BUDDHA INSTITUTE OF TECHNOLOGY
Gida Gorakhpur

Department- Computer Science and Allied
(CSE AND DS)

Program & Semester- B.Tech 3rd Year (6th Semester)

Course and Code- BIG Data Analytics BCDS 061 & BCS 061

Course Outcome

CO No.	Course Outcome	Bloom's Knowledge Level (KL)
CO 1	Demonstrate knowledge of Big Data Analytics concepts and its applications in business.	K1, K2
CO 2	Demonstrate functions and components of Map Reduce Framework and HDFS.	K1, K2
CO 3	Discuss Data Management concepts in NoSQL environment.	K6
CO 4	Explain process of developing Map Reduce based distributed processing applications.	K2, K5
CO 5	Explain process of developing applications using HBASE, Hive, Pig etc.	K2, K5

UNIT-2
HADOOP & MAP REDUCE

Q1. Explain Hadoop Architecture in detail. (AKTU 2022, GATE – Distributed Systems Concept)

Introduction

Hadoop is an open-source framework designed to store and process large datasets using distributed computing. It provides a scalable and fault-tolerant architecture that allows data to be processed across clusters of computers. Hadoop is widely used in Big Data environments and is an important topic in AKTU examinations as well as GATE under distributed systems.

Hadoop Architecture Components

Hadoop architecture mainly consists of two core components namely HDFS (storage layer) and MapReduce (processing layer).

1. Hadoop Distributed File System (HDFS)

HDFS is responsible for storing large datasets across multiple machines in a distributed manner.

Components of HDFS

NameNode

The master node that manages metadata and file system operations

DataNode

Worker nodes that store actual data blocks

Secondary NameNode

Performs checkpointing and assists NameNode

Characteristics

Data is divided into blocks and stored across nodes

Provides fault tolerance through replication

Supports high throughput

2. MapReduce Framework

MapReduce is used for processing large datasets in parallel.

Phases

Map Phase processes input data and generates key-value pairs

Reduce Phase aggregates intermediate results

Characteristics

Parallel processing

Scalability

Fault tolerance

3. YARN (Yet Another Resource Negotiator)

YARN manages resources and schedules tasks in Hadoop clusters

Components

Resource Manager

Node Manager

Functions

Allocates resources

Manages job execution

Working of Hadoop Architecture

Data is stored in HDFS across multiple nodes

MapReduce processes data in parallel

YARN manages resources and scheduling

Advantages

Scalable and flexible

Cost effective

Fault tolerant

Q2. Explain Hadoop Distributed File System (HDFS) Architecture. (AKTU 2021, GATE – Distributed File Systems Concept)

Introduction

Hadoop Distributed File System (HDFS) is a distributed storage system designed to store large datasets across multiple machines. It provides high reliability scalability and fault tolerance which makes it suitable for Big Data applications. HDFS is a core component of Hadoop architecture and is widely covered in AKTU and GATE examinations.

Architecture of HDFS

HDFS follows a master-slave architecture consisting of one master node and multiple slave nodes.

Components

1. NameNode

The NameNode is the master server that manages the file system namespace and metadata

Functions

Maintains directory structure

Stores metadata such as file names and block locations

Handles client requests

2. DataNode

DataNodes are worker nodes that store actual data blocks

Functions

Store data blocks

Perform read and write operations

Communicate with NameNode

3. Secondary NameNode

Assists the NameNode by periodically merging metadata

Functions

Checkpointing

Reducing load on NameNode

Working of HDFS

Files are divided into blocks (default 128 MB)

Blocks are distributed across DataNodes

Each block is replicated for fault tolerance

NameNode keeps track of block locations

Key Features

Fault Tolerance

Achieved through replication of data blocks

High Throughput

Optimized for large data access

Scalability

Can handle increasing data by adding nodes

Advantages

Reliable storage

Cost effective

Handles large datasets efficiently

Q3. Explain MapReduce programming model in detail. (AKTU 2020, GATE – Parallel & Distributed Computing Concept)

Introduction

MapReduce is a programming model used for processing large datasets in a distributed environment. It divides the processing task into smaller sub-tasks and executes them in parallel across multiple nodes. This model is a core component of Hadoop and is widely used in Big Data analytics.

Phases of MapReduce

1. Map Phase

Input data is divided into smaller chunks and processed independently

Function

Generates key-value pairs

Example

Word count where each word is mapped with count 1

2. Shuffle and Sort Phase

Intermediate data is grouped and sorted based on keys

Function

Transfers data between Map and Reduce phases

3. Reduce Phase

Aggregates intermediate results to produce final output

Function

Combines values associated with same key

Working of MapReduce

Input data is split into blocks

Map tasks process each block

Intermediate results are shuffled and sorted

Reduce tasks generate final output

Advantages

Parallel processing

Scalability

Fault tolerance

Limitations

High latency

Not suitable for real-time processing

Q4. Explain the working of HDFS read and write operations. (AKTU 2022, GATE – Distributed File Systems Concept)

Introduction

Hadoop Distributed File System (HDFS) provides a reliable mechanism for storing and retrieving large datasets in a distributed environment. The read and write operations in HDFS are designed to ensure high throughput fault tolerance and data reliability. Understanding these operations is essential for analyzing how data flows in Hadoop systems.

HDFS Write Operation

When a client wants to write data into HDFS the following steps occur

Client sends a request to NameNode to create a file

NameNode checks permissions and availability

File is divided into blocks

NameNode selects DataNodes for storing blocks

Client writes data to first DataNode

Data is replicated to other DataNodes

Acknowledgment is sent back to client

Characteristics

Data is written sequentially

Replication ensures fault tolerance

High reliability

HDFS Read Operation

When a client wants to read data from HDFS the following steps occur

Client sends request to NameNode

NameNode provides block locations

Client directly reads data from nearest DataNode

Data is fetched block by block

If a node fails data is read from replica

Characteristics

High throughput

Fault tolerance

Efficient data access

Advantages

Reliable data storage

Efficient data retrieval

Fault tolerant system

Conclusion

HDFS read and write operations ensure efficient and reliable data handling in distributed environments.

Q5. Explain YARN architecture in Hadoop. (AKTU 2021, GATE – Distributed Systems Concept)

Introduction

YARN (Yet Another Resource Negotiator) is a resource management layer in Hadoop that manages cluster resources and schedules tasks. It separates resource management from data processing enabling efficient utilization of cluster resources.

Components of YARN

1. Resource Manager

Global resource manager of the cluster

Functions

Allocates resources

Manages scheduling

2. Node Manager

Runs on each node

Functions

Manages node resources

Monitors containers

3. Application Master

Responsible for managing application execution

Functions

Negotiates resources

Monitors tasks

4. Container

Unit of resource allocation

Characteristics

Contains CPU memory and other resources

Working of YARN

Client submits application

Resource Manager assigns Application Master

Application Master requests resources

Node Managers execute tasks

Results are returned to client

Advantages

Efficient resource utilization

Scalability

Supports multiple processing models

Conclusion

YARN plays a crucial role in Hadoop by managing resources and scheduling tasks efficiently.

Q6. Explain fault tolerance in Hadoop. (AKTU 2020, GATE – Distributed Systems Reliability Concept)

Introduction

Fault tolerance is the ability of a system to continue functioning even when some components fail. In Hadoop fault tolerance is achieved through data replication and task re-execution ensuring reliable data processing.

Mechanisms of Fault Tolerance

1. Data Replication

Each data block is replicated across multiple DataNodes

Benefit

Ensures data availability even if a node fails

2. Heartbeat Mechanism

DataNodes send regular signals to NameNode

Purpose

Detect node failures

3. Task Re-execution

Failed tasks are automatically reassigned

Benefit

Ensures job completion

4. Checkpointing

Metadata is periodically saved

Benefit

Prevents data loss

Advantages

High reliability

Continuous operation

Error recovery

Conclusion

Fault tolerance is a key feature of Hadoop that ensures system reliability and data safety.

Q7. Explain Hadoop Ecosystem with its components. (AKTU 2022, GATE – Distributed Systems & Data Tools Concept)

Introduction

The Hadoop ecosystem consists of various tools and frameworks that work together to provide storage processing data management and analytics capabilities. These components extend Hadoop functionality and enable efficient Big Data processing.

Major Components

1. HDFS

Distributed storage system

2. MapReduce

Processing framework

3. YARN

Resource management

4. Hive

Data warehouse tool

5. Pig

Data processing tool

6. HBase

NoSQL database

7. Sqoop

Data transfer tool

8. Flume

Data ingestion tool

Characteristics

Scalable

Flexible

Fault tolerant

Applications

Data analytics machine learning real-time processing

Conclusion

Hadoop ecosystem provides a complete solution for Big Data processing.

Q8. Explain difference between HDFS and traditional file system. (AKTU 2019, GATE – File Systems Concept)

Introduction

HDFS is designed for distributed storage of large datasets while traditional file systems are designed for local storage. Both differ in architecture scalability and performance.

Key Differences

Architecture

HDFS uses distributed architecture while traditional systems are centralized

Data Size

HDFS handles large datasets while traditional systems handle smaller data

Fault Tolerance

HDFS uses replication while traditional systems have limited fault tolerance

Scalability

HDFS scales horizontally while traditional systems scale vertically

Performance

HDFS provides high throughput while traditional systems provide low latency

Q9. Explain job scheduling in Hadoop. (AKTU 2021, GATE – Operating Systems & Distributed Systems Concept)

Introduction

Job scheduling in Hadoop determines how tasks are assigned and executed in a cluster. Efficient scheduling improves performance and resource utilization.

Types of Schedulers

1. FIFO Scheduler

Jobs executed in order of arrival

2. Fair Scheduler

Resources distributed equally among jobs

3. Capacity Scheduler

Resources allocated based on capacity

Working

Jobs are submitted

Scheduler assigns resources

Tasks are executed

Results are returned

Q10. Explain data locality in Hadoop. (AKTU 2020, GATE – Distributed Systems Optimization Concept)

Introduction

Data locality refers to processing data near its storage location to reduce network overhead and improve performance.

Concept

Move computation to data instead of data to computation

Types

Node locality

Rack locality

Advantages

Reduced network traffic

Faster processing

Efficient resource usage

Q11. Explain combiner function in MapReduce. (AKTU 2019, GATE – Parallel Processing Optimization Concept)

Introduction

Combiner is an optional function in MapReduce that reduces the amount of data transferred between Map and Reduce phases.

Working

Acts as mini reducer

Processes intermediate data

Reduces data size

Advantages

Improves performance

Reduces network traffic

Q12. Explain difference between MapReduce and Spark. (AKTU 2022, GATE – Distributed Systems & Big Data Processing Concept)

Introduction

MapReduce and Spark are both Big Data processing frameworks but differ in performance architecture and processing techniques.

Key Differences

Processing

MapReduce uses disk-based processing while Spark uses in-memory processing

Speed

Spark is faster than MapReduce

Complexity

MapReduce is complex while Spark is easier

Real-time Processing

MapReduce does not support real-time processing while Spark does

Conclusion

Spark is more efficient for modern applications while MapReduce is suitable for batch processing.